VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

Absolvování individuální odborné praxe

Individual Professional Practice in the Company

2011 Radek Čegan

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Radek Čegan**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: Tieto Czech s.r.o.
2. Struktura závěrečné zprávy:
   a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta.
   b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti.
   c) Zvolený postup řešení zadaných úkolů.
   d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe.
   e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe.
   f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení.

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vede odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Ježek, Ph.D.**

Konzultant bakalářské práce: Mgr. Ondřej Dorazil

Datum zadání: 19.11.2010

Datum odevzdání: 06.05.2011

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: 6.května 2011 .....

Rád bych poděkoval za příležitost praxe ve společnosti Tieto Czech, s.r.o a celému pracovnímu týmu, že mi byli nápomocni a že mi vždy rádi pomohli.

Abstrakt

Tato práce zachycuje průběh vykonávání mé praxe ve společnosti Tieto Czech, s.r.o. V úvodu přibližuji historii společnosti a přináším o ni základní informace. Po úvodu představím prostředí, ve kterém jsem se pohyboval a nástroje, které jsem používal. Poté následují některé typické problémy, které jsem v rámci své praxe musel řešit. Nejprve stručně popíšu zadání – problém a pak uvádím, jak jsem daný problém řešil. Dále uvádím informace o absolvovaném dvoudenním školení, které jsem ve firmě absolvoval a znalosti, které jsem díky němu získal.

Klíčová slova

Unix, Linux, HP-UX, Solaris, AIX, Tieto, praxe, administrace, podpora, diagnostika, LVM, Raid

Abstract

This work shows course of my practice at the company Tieto Czech, s.r.o. At the beginning of this work I am writing about Tieto's history and I am mentioning some basic facts about this company. After that I show you the enviroment I worked in and some tools I worked with. Then follow some common problems I faced during my practice at Tieto. First, I briefly describe the problem and then I show the way I was solving this problem. Next I am mentioning informations about two-day training I was given in the company and knowledge I have gained thanks to it.

Key Words

Unix, Linux, HP-UX, Solaris, AIX, Tieto, practice, administration, support, diagnostics, LVM, Raid

Obsah

1 Úvod

Společnost Tieto[2] je mezinárodní firma čítající přes 17 000 zaměstnanců z přibližně 30 zemí a je to jeden z největších poskytovatelů IT služeb v Evropě. Společnost byla původně založena roku 1968. Vznikla ve Finsku, konkrétně v Espoo, což je součást metropolitní oblasti Helsinki. Původní jméno společnosti bylo Tietotehdas Oy. V roce 1995 byla společnost přejmenována na TT Tieto. O 3 roky později znovu měnila své jméno, a to již na Tieto. Roku 1996 výrazněji pronikla do oblasti telekomunikací. To bylo jistě důležité pro tým, ve kterém jsem vykonával svou praxi. Telekomunikační společnosti totiž hrají mezi zákazníky, jejíchž servery má na starosti významnou roli.
V roce 1999 nastalo historicky významné spojení finského Tieta se švédským Enatorem, čímž došlo opět ke změně jména na TietoEnator. Od března roku 2009 nese společnost jméno Tieto Corporation.
Tieto poskytuje své služby v oblastech finančních služeb, automobilismu, telekomunikací, médií, zdravotní a sociální péče, lesnictví, energií, výroby, maloobchodu a logistiky a veřejnosti.

Svou praxi jsem vykonával v oddělení správy a technické podpory se zaměřením na Unixové systémy. Monitorují se zde především serverové systémy, a to zejména produkční servery Tieta a servery nordických telekomunikačních společností.
V rámci sítě, která je nazývaná BITS (Business ICT Transfer Sweden) se spravují ale také koncové pracovní stanice, popř. servery v této síti.
Tým úzce spolupracuje s jinými týmy, jako např. se síťovým týmem, oddělením zabývajícím se zálohami, nebo aplikačními skupinami. To vše proto, neboť systém je takový základní kámen všeho. Proto se na Unix team mnohdy obracejí se svými požadavky, k jejichž vyřešení je třeba nějaká investigace v systému, oprava, nebo případně instalace nové služby.

2 Prostředí

2.1 Jump Point

Z důvodu zvýšení bezpečnosti zákazníků společnosti Tieto neprobíhá žádná vzdálená správa přímo. Jednak proudí veškerá komunikace zabezpečeným VPN tunelem a navíc se na servery přistupuje srkze tzv. *Jump Point*, což je speciální server v síti Tieto, přes který se dále připojuje buď na další *Jump Point*, nebo na cílový server.

2.2 ITIL

V hierarchii společnosti je nasazen ITIL model, který se používá pro životní cykly jednotlivých požadavků, ať už automaticky generovaných, nebo těch manuálních. Důvod, proč je tento model nasazen je poměrně jednoduchý – Tieto je velká společnost, a proto je třeba, aby byl nasazen nějaký efektivní způsob, jak poskytovat kvalitní a rychlé služby.

2.2.1 Service Desk (SD)

Service Desk se stará o komunikaci s koncovými zákazníky. Ti se na něj mohou obracet se svými problémy, a to buď telefonicky nebo elektronicky prostřednictvím služby zvané mySupport.

2.2.2 Control Desk (CD)

Control Desk se od Service Desku výrazně neliší. Zatímco Service Desk ale zpracovává požadavky od lidí, Control Desk zpracovává ty, které odešle nějaký automatizovaný systém, který dané servery nebo i pracovní stanice sleduje. Tento monitorovací systém bývá nejčastěji software Patrol společnosti BMC.
Část problémů jsou schopni zároveň i sami vyřešit. Pokud jej prověří a zjistí, že se jednalo např. o chybnou detekci nebo pokud se jednalo o problém vznikající z chvilkového přetížení, kdy právě v okamžiku měření může měřená hodnota dočasně překročit práh pro hlášení problému.

2.2.3 Incident Management (IM)

Incident je definován jako *jakákoliv událost, která není součástí běžného provozu a která způsobuje, nebo může způsobit výpadek nebo omezení služeb.*

2.2.4 Problem Management (PM)

Cílem Problem Managementu je předcházet vzniku událostí, které povedou ke vzniku incidentu, případně se zde řeší už vzniklé incidenty. Úzce souvisí s dlouhodobějším monitorováním a databází minulých řešení. Pokud se objevují některé incidenty častěji, může se zde najít permanentní řešení, které sníží celkový počet incidentů, čímž sníží množství potřebných zásahů a tím zefektivňuje práci.

2.2.5 Change Management (CM)

Pokud není opakující se incident řešitelný ani v rámci Problem Managementu, popř. pokud zjistíme, že je problém způsoben např. hardwarovým problémem, je třeba celé řešení eskalovat

na úroveň Change Managementu, kdy je vytvořen požadavek např. na výměnu daného zařízení. Change Management se ale týká i softwaru, např. instalace celého systému.

Samotná *change*, požadavek na změnu, je často rozdělena na několik Action Request. Konkrétně např. zmíněnou instalaci je třeba provést napříč různými týmy. Každý tým poté dostane přiděleny odpovídající požadavky, které dohromady vedou ke splnění vstupního požadavku, což v tomto konkrétním případě byla instalace nového serveru.

2.2.6 Action Requests (AC)

Je konkrétní akce, zaměřena na tým speializovaný v určitém odvětví a většinou je součástí nějakého většího celku v rámci Change Managementu.

2.3 Víceúrovňová podpora

Jednotlivé stupně jsou označovány anglicky jako *Tier*.

2.3.1 Tier 1

První stupeň technické podpory obsahuje ty týmy, které jako první zpracovávají příchozí požadavky klientů. Konkrétně se tedy jedná o Control Desk a Service Desk. Na tomto stupni jsou příchozí požadavky částečně řešeny, ale hlavně jsou třízeny a posílany dále těm správným týmům, které je poté řeší. Navíc mohou odfiltrovávat neoprávněné požadavky. Pokud požadavek vyžaduje nějaký stupeň pověření, případně povolení, provedou hned kontrolu a kontaktují zodpovědná místa, zda je obsah žádosti v pořádku.

2.3.2 Tier 2

Druhý stupeň se již stará o odborné řešení problémů a je zde posíláno vše, co nemohlo být jednoduše vyřešeno na prvním stupni. Očekává se zde potřebná technická odbornost a specializace. Zároveň si ale technici z 2. stupně projdou práci, která byla vykonána již na prvním stupni, aby nebyla provedena stejná činnost vícekrát a aby např. zákazníkovi nebyl položen tentýž dotaz vícekrát. Tyto informace personál prvního stupně přiloží k ticketu. Přiložené informace mohou obsahovat přesnější instrukce, než původní požadavek zákaznika, který mohl být neúplný.

2.3.3 Tier 3

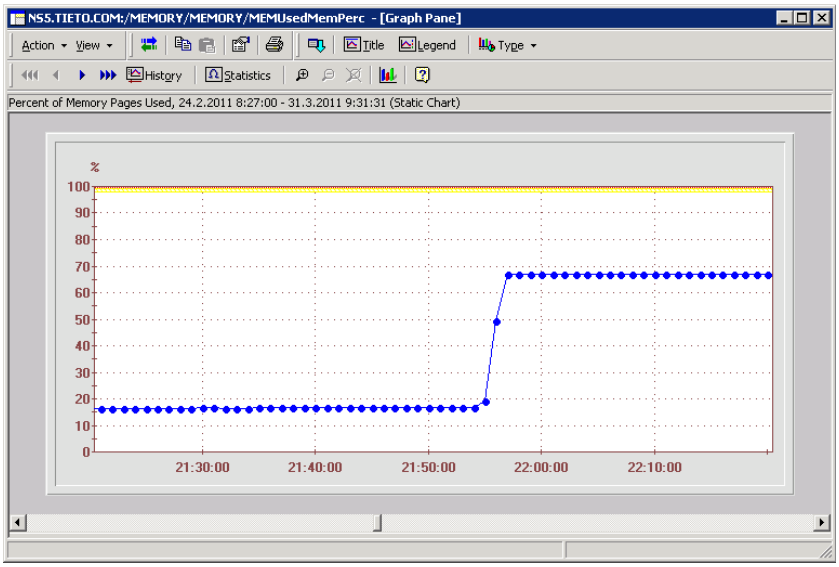
Tento stupeň je poslední ze základního modelu víceúrovňové podpory. Jsou zde předávány problémy, které nemohly být úspěšně vyřešeny ani na druhém stupni. Důvod k nemožnosti korektního vyřešení přitom nemusí být jen nenalezení vhodného řešení. S rozdělením práce podle odbornosti totiž přichází rozdělení množství přidělených oprávnění. Zaměstnanci např. Control Desku totiž obvykle nepotřebují mít rozsáhlé množství pravomocí, protože problémy, k jejichž řešení jsou potřebné, neřeší. Obdobný případ může nastat i u Tier 2, kteří jsou nuceni problém předat svým kolegům jen z důvodu omezených pravomocí.

Tým, ve kterém jsem vykonával svou praxi figuruje většinou jako Tier 2, ale neustále přibývá serverů, kde je zodpovědný jako Tier 3.

3 Využívané nástroje

3.1 BMC Patrol

Patrol je monitorovací systém, který sestává ze 3 částí: PatrolAgent, konzolový server a Patrol Central. PatrolAgent je serverová část řešení. Nachází se na každém monitorovaném serveru, odkud odesílá data konzolovému serveru. S ním poté komunikuje klientská část systému, kterou je Patrol Central.

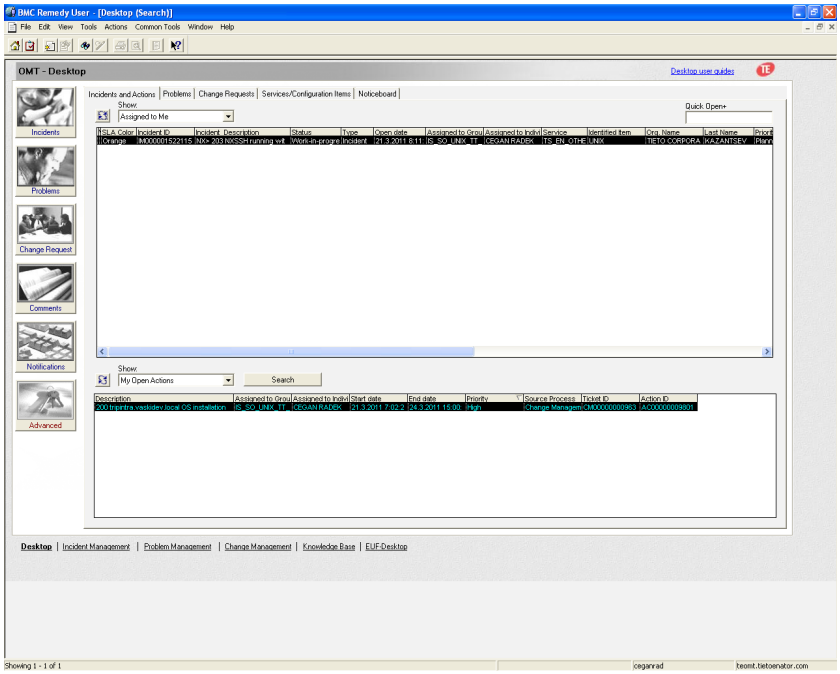


Obrázek 1: Monitorovací systém Patrol Central

Patrol Central slouží k zobrazování nasbíraných dat z konzolového serveru, kde se data schraňují. Monitorují se veškerá relevantní data, jako je využití pevných disků, pamětí, swapu, sítí, připojených konektorů, procesoru, logu, procesy nebo třeba podstatné atributy databází.

3.2 BMC Remedy

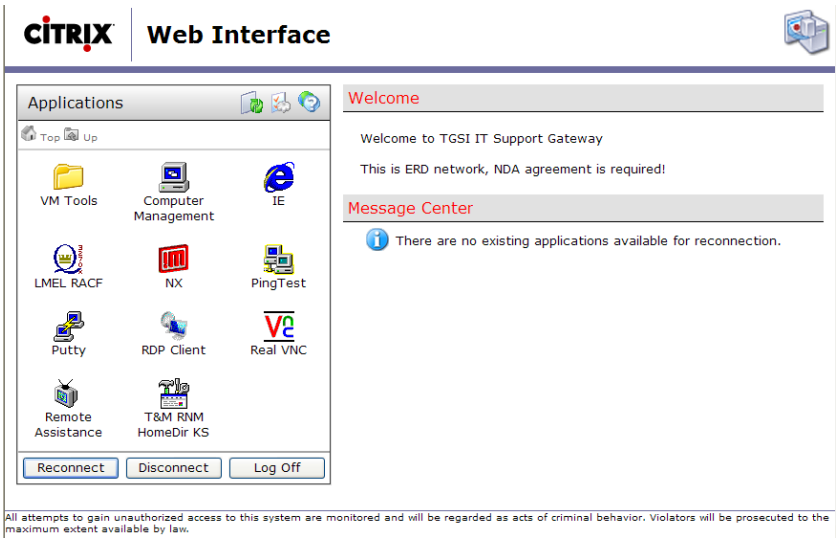
Tento software slouží pro zobrazení požadovaných ticketů a umožňuje s nimi manipulovat. Je přímo stavěn do prostředí, která implementují ITIL model, proto je s ním tento program plně kompatibilní. Remedy tedy umožňuje kromě zadávání řešení ticketů zobrazovat již provedenou práci(*WL – Work Log*), dá se vyčíst, kudy ticket putoval a který technik na něm v minulosti pracoval, díky čemuž je jej snadné kontaktovat. Na základě známého popisu problému je možné vyhledávat v historii jemu podobných, což může taky urychlit celý proces, protože z jejich řešení může být např. patrné, že se problém již řeší v rámci jiného ticketu.



Obrázek 2: ITIL nástroj pro práci s tickety Remedy

3.3 Citrix klient

Tento PlugIn do webového prohlížeče od společnosti Citrix Systems je použit pro přístup do sítě BITS. Umožňuje spuštění aplikace PuTTY, která slouží pro vzdálený přístup přes SSH, internetového prohlížeče, Ping Test, který je užitečný pro zjištění, zda systém běží. Užitečná je i aplikace NX Client, která umožňuje připojení ke vzdálené ploše linuxového systému skrze SSH tunel.



Obrázek 3: Webový PlugIn Citrix

4 Typické problémy

4.1 Plný disk

Jelikož jedna ze zásad jsou pravidelné zálohy, tak se často stane, že se disk skoro zaplní, poté je třeba buď odrotovat staré logy, nebo hledat, co je možno buď smazat, nebo archivovat. Samotné smazání je pochopitelně primitivní záležitost. Problém může být mezi desítkami GB dat najít ta "nepotřebná", nehledě na to, že tyto akce je nutno provádět jako superuživatel a je tedy na místě vysoká míra opatrnosti. Ne vždy navíc na discích bývá postradatelný obsah, na více používaných systémech se na ně neustále zapisuje nemalé množství dat. Pokud již není co smazat, je nutné kontaktovat zákazníka a nabídnout mu zvětšení disku, což se poté zpracovává ve vlastním ticketu. Nejobjemnější soubory bývají navíc data typu Oracle databáze, k čemuž ani nejsme kompetentní a tyto problémy pak musí vyřešit databázový tým.
Nejobjemnější soubory na souborovém systému je možno zobrazit např. tímto příkazem.

```
du -akx | sort -rn | more
```

Následující příkaz provede archivaci všech souborů, které budou ve svém názvu obsahovat *log* a zároveň nebudou mít příponu *gz*, čímž se zamezí komprimaci již komprimovaných souborů. Tímto příkazem budou navíc ovlivněny jen soubory starší než 7 dní.

```
find ./*log* -type f ! -name \*\.gz -mtime +7 -exec gzip -v9 {} \;
```

4.2 Instalace nového serveru

Také instalace můžeme provádět vzdáleně. Pokud se jedná o virtualizovaný stroj, poté se správa včetně instalace provádí přes VMware Vcenter. Sám jsem tyto instalace také prováděl. Server jsem si zkusil nainstalovat jen abych se seznámil s prostředím VMware Vcenter a později i v rámci požadavku na instalaci produkčního serveru. Vzdáleně může Unix team instalovat i dedikované servery na platformě HP Blade. Tato instalace se provádí pomocí speciálního rozhraní v Javě a běžícího v prohlížeči.

4.2.1 Příklad

Poté, jakmile je dokončena instalace, provede se vstup do *init 3*, čímž dojde k zakázání případného *X Serveru*. Toto se též nastaví v konfiguračním souboru */etc/inittab*, kde se upraví následující řádka.

```
d:3:initdefault:
```

Taktéž je zde třeba zakázat zachycení kláves CTRL+ALT+DEL, což by provedlo reboot systému.

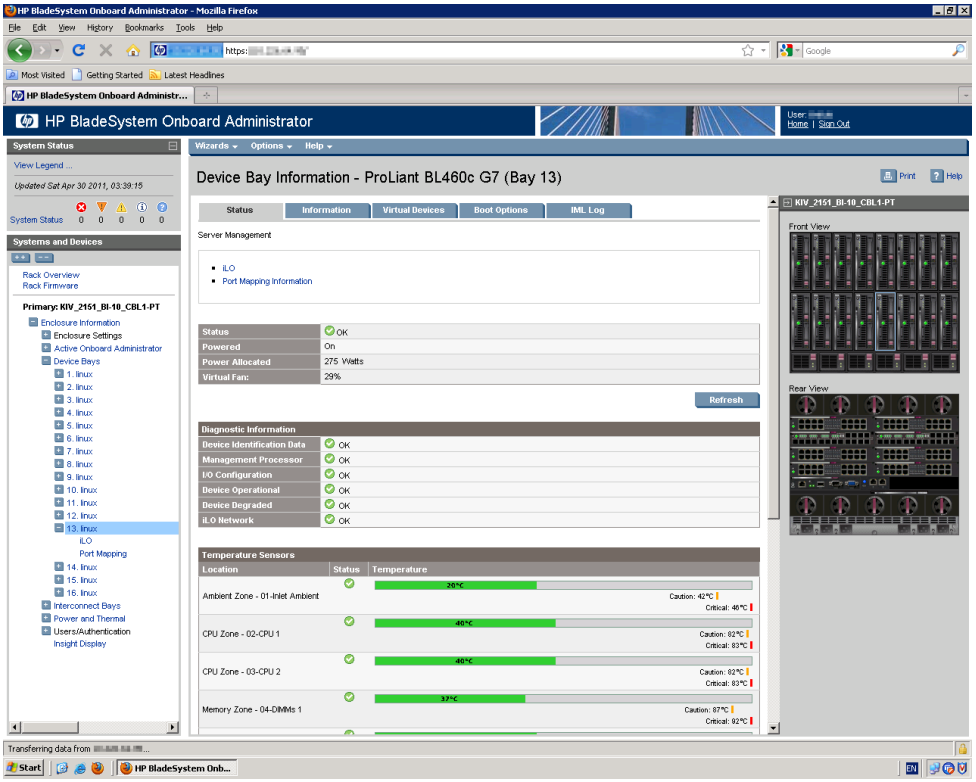
```
#ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

Dále se provede editace souboru */etc/sysconfig/network*, kam se zapíše výchozí brána.

```
GATEWAY=x.x.x.x
```

Následuje konfigurace síťových rozhraní. Na serverech se obvykle používá *BOND*, pro případ, že by se jedno rozhraní poškodilo, aby jej nahradilo jiné. Proto je třeba nejříve editovat soubor */etc/modprobe.conf*, kde se přidají tyto řádky, čímž dojde k nastavení modulu, který *BOND*ing zajišťuje.

```
alias bond0 bonding
options bond0 miimon=100 mode=active-backup primary=eth5
options hangcheck-time hangcheck_tick=30 hangcheck_margin=180
```



Obrázek 4: HP BladeSystem Onboard Administrator

Nyní je možné přistoupit ke konfiguraci jednotlivých síťových rozhraní. Ty bývají alespoň 3. Dvě z nich jsou produkční a jedna pro provádění síťových záloh. Konfigurace *BOND*u vypadá následovně.
Jako první virtuální síťové rozhraní reprezentující *BOND*.

```
DEVICE=bond0
BOOTPROTO=none
ONBOOT=yes
IPADDR=x.x.x.x
NETMASK=255.255.255.0
```

Nyní první fyzické rozhraní, které je součástí *BOND*u

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
HOTPLUG=no
MASTER=bond0
SLAVE=yes
```

Druhé fyzické rozhraní, které začne pracovat v případě poruchy prvního fyzického rozhraní.

```
DEVICE=eth1
BOOTPROTO=none
ONBOOT=yes
HOTPLUG=no
MASTER=bond0
SLAVE=yes
```

Nyní, když je funkční síťové připojení, je možno produkt zaregistrovat na webových stránkách *www.redhat.com/register*, kde dojde k aktivaci licence. Tu poté následovně vložíme do systému.

```
rhnreg_ks --profilename "HOSTNAME" --username "redhat_login" \
--password "********" --proxy "x.x.x.x:3128" \
--subscription="XXXX-XXXX-XXXX-XXXX"
```

Jakmile je licence spárována s Redhat servery, je možno server aktualizovat.

```
yum update
```

Nyní je ještě třeba nainstalovat zálohovací službu a monitorovacího agenta, případně podle požadavku vytvořit účty pro zadavatele instalace.

4.3 Definice VLAN

Přicházejí požadavky na vytvoření základního síťového prostředí před instalací samotného systému. Samotná tvorba VLAN má 2 kroky, ten první udělá networking tým, kdy VLANy zpřístupní a protože UNIX tým se stará o správu VMware, je na nás, abychom umožnili k těmto prostředkům přístup z prostředí VMware.

4.4 Ztráta spojení se serverem

Na serverech obvykle běží monitorovací systém Patrol od společnosti BMC. Ten nás upozorní, pokud nedokáže kontaktovat svou klientskou část na daném serveru. Může to znamenat, že server vůbec neběží, nebo že odmítá síťová připojení. Potom je obvykle nutné provést menší výzkum a zjistit, proč tomu tak je. Kontaktovat kolegy v zahraničí, kteří třeba můžou s daným serverem pracovat, zjistit od networking týmu, jestli běží síť a až pokud budeme mít povolení od zodpovědných míst budeme se moci, připojit na iLO rozhraní daného serveru, popř. celého racku a daný server restartovat, případně nastartovat, pokud neběží. A to vše za předpokladu, že pro tuto akci na daném serveru budeme mít oprávnění, jinak nezbude nic jiného, než akci svěřit zodpovědnému týmu.

Ztráta spojení taky ale může znamenat, že jednoduše neběží daný PatrolAgent, který stačí jen nastartovat / restartovat.

4.5 Problém s rozhraním

Může být dočasný problém způsobený náhlou zátěží, ať už celého systému, daného rozhraní, nebo obou faktorů dohromady. V případě dlouhodobějšího problému může být třeba dané rozhraní vyměnit, pokud je poškozeno. Pokud je rozhraní v pořádku, je třeba znovu provést výzkum. Pokud se jedná o síťové rozhraní např. zkontrolovat síť, zkontrolovat nastavení v systému.

Příklad, se kterým jsem se setkal. Na serveru každou chvíli dočasně spadla síť, všechno nastavení bylo v pořádku. Nakonec se ukázalo, že problém byl nejspíš v hardwaru. Byly tam totiž 2 síťové karty, které byly sbondovány v jedno rozhraní a nastaveny v režimu backup, kdy síťová karta nahradí druhou při problému. Vada byla asi v jednom páru kabelu na primární síťové kartě, protože při pingu se pakety občas vracely jako duplicitní, což byla chvíle, kdy se přepínalo na záložní rozhraní.

4.6 Insmod error

Ze serveru přišla chybová hláška vygenerovaná jádrem operačního systému. Ta zněla

insmod: Hint: insmod errors can be caused by incorrect module parameters including invalid IO or IRQ parameters. You may find more information in syslog or the output from dmesg.

Insmod slouží pro vkládání modulu do běžícího jádra. Zde nám tedy hlásí, že byl některý modul volán se špatnými parametry a že můžeme hledat více informací v systémových log souborech. To jsou koneckonců první místa, kde se dá v jakékoli situaci hledat nějaké vodítko. Nic konkrétního se ale nikde vyčíst nedalo, proto jsem začal zkoumat, kdo co na tomto serveru dělal přede mnou. Linuxový interpret *BASH* si totiž uchovává poslední zadané příkazy do systému, a to do skrytého souboru v uživatelově domovském adresáři pojmenovaném *.bash_history*.

V historii naposledy zadaných příkazů mě zaujal především příkaz *fdisk*, který slouží pro práci s diskovými oblastmi. Už teď jsem nebyl daleko od pravdy, protože původce ony chyby jádra úzce souvisí právě s programem *fdisk*.
Pokud je totiž provedena změna, jako např. rozšíření diskové oblasti (a poté i souborového systému), je třeba znovu načíst tabulku rozdělení disků, aby mohlo s nově rozděleným diskem jádro správně pracovat.

K tomu je užíván program *partprobe*, který se o znovu načtení tabulky postará. Tento program byl také nakonec i spuštěn, ale zprvu jsem mu nevěnoval tolik pozornosti. Při procházení dokumentace tohoto programu jsem potom zjistil, že se snaží projít všechna zařízení, včetně disketové mechaniky, která ale na serveru nebyla fyzicky dostupná, což způsobilo chybný přístup k tomuto zařízení.

4.7 Nedostatek dostupné operační paměti.

Toto ve skutečnosti nebyl nijak závažný problém, i když to tak na první pohled vypadá. Z více serverů chodily zprávy, že je operační paměť využita přibližně na 99%. Pokud by tomu tak skutečně bylo, byl by to určitě závažný problém. Operační systém Linux ale přistupuje ke správě operační paměti jinak, a sice tak, aby byla využita co nejefektivněji. Pokud je v systému dostupná, neobsazená paměť, snaží se tuto paměť alokovat jako cache. Tím dojde k urychlení běhu celého systému, protože se sníží počet diskových operací.
Monitoring byl ale nastaven tím způsobem, že sledoval skutečně volný prostor, což také jistě může být v jistých situacích užitečné. Způsobovalo to ale taky spoustu chybných zpráv o využití paměti.
Operační systém Linux lze donutit, aby tuto cache uvolnil. Celý proces shrňuje tento skript, který jsem si vytvořil.

```
#!/bin/bash
if [ "$(id -u)" != "0" ]; then
  echo "This script must be run as root" 1>&2
  exit 1
fi
sync
sleep 1
echo 1 > /proc/sys/vm/drop_caches
sync
sleep 1
echo 0 > /proc/sys/vm/drop_caches
sync
```

Toto řešení ale není dobré, jednak proto, že je jednorázové a jednak proto, že snižuje výkonostní potenciál systému.
Nejlepší řešení by bylo změnit způsob monitorování, pokud jsem jej chtěl ale zachovat, musel jsem najít jiné vhodné řešení. Toto jsem nalezl a jeho provedení je snadné.

```
echo -n 70000 > /proc/sys/vm/min_free_kbytes
```

Tento příkaz nastaví systém tím způsobem, že se nebude snažit plnit cache, pokud bude volné méně než 70 000 kilobyte prostoru v operační paměti.

5 BITS

5.1 Problém při zápisu na FTP server, chybová hláška, že je plný disk

Disk ovšem plný nebyl. Ani po přímém připojení na daný server se do cílové složky FTP serveru nedalo zapisovat, nejprve jsem hledal problém v oprávnění, ale marně. Po zamyšlení nad tím, jak filesystém funguje na nižší vrstvě jsem odhalil zdroj problému. Filesystém měl vyčerpán všechny dostupné inody. A vzápětí jsem odhalil i proč - na stejném filesystému byla i mailová fronta ve které bylo 335 975 souborů, které spotřebovávaly většinu inodů. Řešení je dost obtížné, prakticky se dá filesystém zvětšit, pokud je kam. Dalším řešením by bylo jedině jej přeformátovat a definovat více inodů.

5.2 Přidání uživatele do mailing listu

V systému BITS se nachází i několik mail listů, konkretní uživatel musel byl přidán do mailing listu, aby mu z něj chodila pošta a aby měl právo přes něj i odesílat.

5.3 Špatně nastavena doména, ve které se mají hledat hostname

Na daném systému se tyto informace nastavovaly přes DHCP, který z nějakého důvodu hodnotu domény nastavil špatně. Tyto věci se pochopitelně nastavují na DHCP serveru. Zjistit IP adresu DHCP serveru v síti, který zaslal nastavení nemusí být úplně jednoduché. Žádosti DHCP REQUEST a poté odpověď DHCP REPLY se totiž posílají broadcastem, takže server může být kdekoli na síti, dokonce jich může být více. V některých unixových systémech se adresa serveru, který přiděloval informace uloží do speciálního souboru, avšak ne na tomto systému. Nakonec jsem si zažádal o přidělení nových informací a během tohoto procesu se adresa naštěstí ukázala. SSH ale na tomto serveru neběželo. Proskenoval jsem si otevřené porty a zaujal mě port 80, na kterém se po připojení přes konzolový prohlížeč objevila defaultní stránka IIS, což znamenalo, že na serveru běží operační systém Windows, takže jsem přesně zjistil, kdo může změnu provést.

5.4 Práva a skupiny

Měl jsem změnit skupinu pro celý adresářový strom na fileserveru a nastavit konkrétní oprávnění přístupu.

5.5 Nový uživatel na FTP server

Úkol byl přidat přístup pro nového uživatele do konkrétní části adresářového stromu s konkrétnimi právy. Problém s uživatelskými účty v BITSu je, že se všude dopropagují účty z NISu. Tento uživatel však potřeboval pouze přístup na konkrétní FTP server, navíc se stejným loginem, jako už měl založen v NISu. Tento problém jsem řešil i se švédským kolegou, který mi potvrdil, že jej tam mohu přidat lokálně. Systém to standardním způsobem ale nedovolí, protože už vidí to jméno z NISu, takže bylo třeba uživatele přidat manuálně editací příslušných konfiguračních souborů. Při přístupu na server, se pak data lokální upřednostní před těmi v NISu.

5.6 Přístup na server přes SSH

Uživatel hlásil problém, že se mu nedaří přes SSH připojit na vzdálený server, ačkoli byl před několika týdny údajně schopen se na něj připojit.
Toto prostředí se nacházelo v Číně a uživatelé byli nuceni k síti přistupovat prostřednictvím proxy serveru, který sloužil jako bezpečnostní prvek. Většinu informací jsem si musel vyžadat přímo od zákazníka. Od něj jsem se např. dozvěděl, že některým jeho kolegům spojení na tento server funguje bez problémů, ti totiž nemuseli používat k přístupu na síť žádný proxy server.
Potom jsem tedy kontaktoval networking tým, od kterých jsem zjistil, že firewall na proxy serveru filtruje spojení na port 22, na kterém SSH Daemon běží. Zajímavé bylo, že tento konkrétní server, na který se zákazník potřeboval připojit, byl dostupný volně z internetu, a připojit se na něj tedy nebylo možné čistě z interních bezpečnostních mechanismů.
Napadlo mě sice řešení, které by se nejspíše dalo aplikovat pomocí správného směrování skrze stanice, které můžou tento server kontaktovat, to by ale bylo v podstatě obcházení bezpečnostních pravidel a tudíž nekorektní. Později jsem se od zákazníka ještě dozvěděl, že si všimnul změny IP adresy proxy serveru, což by se dalo vysvětlit tím, že byl server přeinstalován, nebo nahrazen jiným. Zároveň to vysvětluje náhlou nemožnost kontaktovat server.
Tento problém tedy nebylo možno vyřešit jinak, než tím, že si zákazník musel podat žádost o otevření portu na firewallu proxy serveru.

6 Linux Training

V rámci své praxe ve společnosti Tieto jsem absolvoval také dvoudenní školení od zkušeného školitele, který v minulosti pracoval mimo jiné i na Fakultě elektrotechniky a informatiky Vysoké školy Báňské v Ostravě.

Toto školení bylo pro výuku pokročilejších metod a navazovalo na základnější školící kurs téhož školitele.

Během prvního dne jsme se učili pracovat s počítačovým jazykem AWK pro pokročilou práci s texty, poté jsme si řekli něco o utilitě sysctl, následovala výuka linuxového firewallu iptables a na konci prvního dne jsme se naučili LVM - Logical Volume Management.

Druhý den jsme se věnovali především výuce softwarových diskových Raidů a teoretické výuce bezpečnostního protokolu Kerberos.

6.1 AWK

AWK je počítačový jazyk, který slouží ke zpracovávání textových dat. Jeho použití je všestranné – dá se použít jak na zpracovávání dat uložených v souboru, tak na živé procházení streamů. V unixových systémech je přebírání zdrojových dat ke zpracování nejčastěji řešeno využitím tzv. *roury*, tj. přesměrováním výstupu jednoho procesu do vstupu jiného procesu.

6.1.1 Kontrukce jazyka

Programu AWK jsou předávany dva zdroje dat. Prvním je zdrojový kód, který obsahuje samotné příkazy, které se provedou pro každý vyhovující řadek vstupního textu, což je druhý zdroj dat. Jak samotné instrukce programu, tak zpracovávaná data je možné zadat přímo v konzoli. Také je možné příkazy načíst z konfiguračního souboru a zkoumaná data mohou být v regulérním textovém souboru.

Spouštění programu AWK se řídí následujícím pravidlem

/vzor/ {*akce*}

Vzor je regulární výraz, podle kterého se porovnávají jednotlivé řádky zkoumaného textu. Pro tyto řádky je poté aplikována kýžená akce.
Akcí se pak rozumí sekvence příkazů, které jsou provedeny u odpovídajících řádků.
V příkazech je možno jako u klasických programovacích jazyků použít kupř. přiřazování hodnot do proměnných, vytváření a volání funkcí, nebo i provádění výpočtů. Výpis vybraných a přeformátovaných informací se potom provede pomocí příkazu *print*.
Jazyk obsahuje ještě 2 speciální konstrukce, a sice *BEGIN*, která je volána hned na začátku programu a *END*, která se provede těsně před ukončením programu.

6.1.2 Proměnné

Program AWK si uchovává i některé vnitřní proměnné, které se mohou použít např. k řízení běhu programu.

Záznam

Jako záznam je obvykle považován jeden řádek textového souboru, fakticky se ale jedná o kusy dokumentu oddělené speciálním znakem. Hodnota tohoto znaku se ukládá do interní proměnné **RS**(Record Separator) a standardně je nastavena na oddělovač řádku.
Číslo aktuálně procházeného záznamu je uloženo v proměnné **NR**(Number of Records).

Položka

Položka je samostatný prvek zpracovávaný v rámci záznamu. Jednotlivé položky záznamu od sebe odděluje výraz uložený v proměnné **FS**(Field Separator) a standardně se jedná o bílé znaky, tedy mezery a tabulátory. Počet položek aktuálního záznamu je možno přečíst v proměnné **NF**(Number of Fields).
Načtený záznam si je možno představit jako pole, k jednotlivým položkám se pak přistupuje přes indexy.

$0	Speciální položka – obsahuje celý záznam
$1	První sloupec záznamu
$2	Druhý sloupec záznamu
	atd.

Tabulka 1: Přístup k položce

Pokud je třeba pracovat s názvem zpracovávaného souboru, dá se jeho název najít v proměnné **FILENAME**.

Ve výstupu textu si můžeme sami zvolit i to, jakým způsobem se bude formátovat vypisování záznamů i položek. Vše se opět nastavuje obsahem interních proměnných.
ORS(Output Record Separator) definuje způsob oddělování záznamů a **OFS**(Output Field Separator) říká, jak od sebe oddělit jednotlivé položky.

6.1.3 Příklad

```
ls -l /boot/ | awk 'BEGIN\{s=0} /^[^d]/{s+=$5} END {print s}'
```

Program *ls* slouží k výpisu obsahu adresáře. Přepínačem *-l* zobrazíme více informací o souborech a adresářích na disku. Za ním následuje cesta k místu na disku. Poté následuje roura, kterou takto získané informace předáme na vstup programu *AWK*, pro něj to tedy budou surová textová data k zpracování.
Ihned po spuštění programu dojde k inicializaci proměnné *s*, která je nastavena na hodnotu 0. Následuje vzor v podobě regulárního výrazu, kterým říkame, že má najít všechny takové řádky,

které nezačínají písmenem *d*. Takto ve výpisu programu *ls -l* začínají záznamy značící adresáře, dojde tedy k vybrání pouze souborů. V sekci akce pak dojde k inkrementaci proměnné *s* o hodnotu páteho sloupce, což je velikost souboru v bytech. Až takto budou zpracovány všechny řádky, dojde k volání výpisu proměnné *s*. Výsledek je tedy jen jediná číselná hodnota, která obsahuje velikost všech souborů v bytech v daném adresáři.

```
cat /etc/passwd | awk 'BEGIN {FS=":"} {printf "%5d: %10d\n",NR,$3}'
```

V tomto příkladu dochází k výpisu dat ze souboru, který obsahuje informace o uživatelských účtech. Jednotlivé parametry účtů jsou od sebe odděleny znakem *:*, proto bylo třeba jej nastavit i v programu *AWK* jako oddělovač položek. K výpisu je potom použit příkaz *printf*, který je typický pro jazyk C. Na výstupu bude číslo řádku následované dvojtečkou a hodnotou UID uživatele, který je na daném řádku uložen.

```
#!/bin/bash
prg='$0'
tmp=/tmp/$prg-tmp-$$
dir=${1-.}
echo "Adresar $dir:"

sed -n '/^#START FILE$/,/^#END FILE$/p' $0| sed 1d|\
 sed '$d'|sed '/^#/s///' > $tmp
ls -l $dir |awk -f $tmp
rm -f $tmp
exit 0
#START FILE
#BEGIN {s=0}
#/^[^t]/{s+=$5;f++}
#END{printf "Celkem %d bytu v %d souborech\n",s,f}
#END FILE
```

Tento komplikovanější a komplexnější skript obsahuje jak instrukce pro program *AWK*, tak samotné volání tohoto programu. Příkazy programu *AWK* jsou na konci souboru a jsou zakomentovány, takže je linuxový příkazový interpret ignoruje. Pomocí proudového textového editoru jménem *SED* dochází k procházení souboru a vše mezi statěmi *START FILE* a *END FILE* zkopíruje bez úvodních znaků do dočasného souboru a ten je poté předávan jako konfigurační programu *AWK*.

6.2 Sysctl

Linuxové jádro je poměrně flexibilní, za běhu je do něj možno vkládat různé moduly, nebo pomocí systémové utility jménem *sysctl* taktéž za běhu modifikovat některé jeho parametry a tím měnit způsob, jak se chová.
Parametrů, které se dají takto změnit jsou stovky. Takováto změna se promítne okamžitě, a to bez nutnosti systém restartovat, nebo jádro překompilovávat. Takto nastavené parametry ovlivňují jádro jen jednorázově, dočasně. Pro trvalé nastavení parametrů, které se opět nastaví po restartu systému, je lze zadat do konfiguračního souboru */etc/sysctl.conf*. Právě zde totiž program *sysctl* při startu hledá parametry, které by mohl nastavit.

6.2.1 Příklad

Pro nastavení, aby se systém restartoval např. po 10 vteřinách od kolapsu jádra, stačí nastavit tyto parametry.

```
kernel.panic = 10
kernel.panic_on_oops = 10
```

6.3 LVM – Logical Volume Management

Klasický způsob, kterým je dělen disk na oblasti s sebou nese značné komplikace při manipulaci s nimi. Ty jsou navíc o to markantnější v serverovém prostředí. Do jisté míry to souvisí i s rostoucí popularitou virtualizace, která šetří peníze za hardware, energii, prostory a efektivně využívá potenciál hardwaru, čímž i šetří životní prostředí. V praxi si při instalaci zákazník zaplatí za určité diskové prostory, během životního cyklu serveru se ale může stát, že zakoupený diskový prostor už nemusí být dostatečný.

Pokud by oddíly na serveru byly rozděleny klasickým způsobem, tak se např. redistribuce již zakoupeného prostoru v rámci existujících oblastí stane netriviálním zákrokem. Bylo by zapotřebí jednu fyzickou oblast zmenšit a druhou zvětšit. Vzhledem k tomu, že oblasti leží jedna za druhou, je navíc bezpodmínečně nutné, aby takto modifikované oblasti ležely hned vedle sebe. Navíc tato operace s sebou nese jistá rizika, že by výsledný souborový systém mohl být poškozen.

Tyto problémy se snaží řešit *LVM*. Usnadňuje rozšiřování existujících oblastí, při přidávání nových je efektivně využíváno všechno dostupné místo ze všech fyzických disků. Umožňuje i např. vyjmutí fyzického disku při zachování všech dat i rozdělení virtuálních disků, a to za předpokladu, že na jiných discích je dostatek místa – dojde k přesunu všech dat z disku jinam. *LVM* navíc umožňuje vytvořit tzv. *snapshot*, což je obraz celého oddílu. Jeho výhodou je, že se dá vytvořit za běhu a je neměnný, což se dá využít např. pro zálohy, jelikož nedochází ke změnám dat. Ukázku tvorby snapshotů a více podrobností lze najít ve stručném přehledu *LVM* na portálu Root.cz[5].

6.3.1 Hierarchie

Fyzický disk

Úplně ta nejnižší vrstva. Samotné hardwarové zařízení, které se připojuje do počítače.

Logický disk

Vytvoření nenaformátované oblasti na fyzickém disku. Až do tohoto místa je logika stejná, jako při běžném přístupu k diskovým oblastem.

Fyzický oddíl (*Physical Volume – PV*)

Fyzický oddíl se vytváří nad logickým diskem a slouží pro přípravu disku k použití pod *LVM*. Takto připravený disk se potom bude v systémech, které nejsou schopny pracovat s *LVM* jevit jako prázdný a bez oddílu a při pokusu přistoupit na něj se mohou snažit disk naformátovat.

Skupina oddílů (*Volume Group – VG*)

Skupina oddílů sjednocuje jeden nebo více Fyzických oddílů do jednoho celku. Tento celek je základ pro vytváření jednotek, na která bude možno ukládat data. Prakticky se jedná o *pool* dostupného místa, ze kterého je poté možno jednotky vytvářet nebo zvětšovat.

Logický oddíl (*Logical Volume – LV*)

Logický oddíl je již posledním stupněm v hierarchii *LVM*. Jedná se o konečné blokové zařízení, které bude možno formátovat a vytvořit na něm souborový systém, se kterým bude možno pracovat.

6.3.2 Příklad

Tímto se oddíl připraví pro použití s *LVM*

```
pvcreate /dev/sda1
```

Ověření, že byl oddíl přidán jako fyzický oddíl zobrazením všech takovýchto oddílů v systému.

```
pvs
```

Vytvoření skupiny oddílu sestávající z jediného oddílu.

```
vgcreate MyVolGrp0 /dev/sda1
```

Zobrazení dostupných skupin oddílů.

```
vgs
```

Vytvoření logické jednotky o velikosti 3 gigabyte. Tímto se vytvoří blokové zařízení v /dev/MyVolGrp0/jmeno

```
lvcreate -L 3G -n jmeno MyVolGrp0
```

Výpis všech logických jednotek.

```
lvs
```

Nyní je vytvořena jednotka, se kterou je možno pracovat stejně, jako přímo s logickým diskem. Následuje tedy vytvoření souborového systému.

```
mke2fs -j /dev/MyVolGrp0/jmeno
```

Největší předností *LVM* je možnost rozšířit kapacitu disku. To se provede obdobně jako jeho vytvoření. Rozšíření o 9 gigabyte by vypadalo následovně.

```
lvextend -L +9G /dev/MyVolGrp0/jmeno
```

Takto se provedlo rozšíření jen logické jednotky. Souborový systém zůstal ve stejné oblasti a velikosti, proto je jej nutné roztáhnout přes celou jednotku.

```
resize2fs /dev/MyVolGrp0/jmeno
```

6.4 Softwarové RAIDy

Softwarovým *RAID*em se rozumí provozování *RAID*u, který je řízen systémem a ne samotnými disky na nižší úrovni. Je takto možné např. zapojit do *RAID*u i *LVM* disky. Různých typů *RAID*ů existuje daleko více, než kolik jich budu zmiňovat. Vyberu totiž jen ty hlavní, které mají podporu v implementaci v Linuxu. Kompletní přehled je možno nalézt např. v on-line encyklopedii[3].
V Linuxových systémech slouží pro práci s *RAID*y program *mdadm*, konfigurace se ukládá do souboru */etc/mdadm.conf* a informace o *RAID*u v systému lze najít v */proc/mdstat*.

Raid 0 (stripe)

Toto není úplně plnohodnotný *RAID*, protože nedochází k ochraně dat před poškozením disku. Jedná se o skládání kapacity disků do jednoho, přičemž přenos probíhá mezi disky střídavě po stopách. Tento způsob přístupu k disku urychluje čtení i zápis, jelikož umožní pracovat s daty na obou discích současně.

Raid 1 (mirror)

Použitím zrcadlení dochází k uchovávání dat na dvou discích. Tento způsob ochrany dat nefiguruje jako záloha, pouze chrání před poškozením disku. Výhodou může být vyšší rychlost čtení a nižší přístupová doba k datům, zápis ale může být pomalejší.

Raid 4

Počet disků v poli je navýšen o 1. Tento disk slouží jako paritní, na který jsou ukládany kontrolní součty dat ze všech disků, a to po blocích. Nevýhodou je, že tento disk může být slabým místem celého řešení, právě z důvodu, že slouží k ukládání parit ze všech disků.

Raid 5

Jedná se o obdobu Raid 4 s tím rozdílem, že parita je postupně zapisována na všechny disky.

6.4.1 Příklad

```
mdadm /dev/md0 –C –l 0 –n 2 /dev/sda1 /dev/sdb1
```

Tento příkaz vytvoří v systému nové *RAID 0* pole složené ze 2 jmenovaných disků.

```
mdadm --detail --scan > /etc/mdadm.conf
```

Zadáním tohoto příkazu dojde k uložení běžícího nastavení do konfiguračního souboru.

7 Shodnocení praxe

7.1 Uplatněné znalosti získané v průběhu studia

Na Vysoké škole báňské se sice mnoho předmětů zaměřených na linux nevyučuje, přesto jsem některé znalosti určitě využil. Nejvíce prospěšným shledávám předmět Počítačové sítě. V tomto předmětu probíhala výuka v operačním systému Linux. Znalosti týkající se problematiky Linuxu jsem sice již všechny znal, využil jsem ale získané vědomosti ze síťové oblasti. Správa serverů do značné míry souvisí se síťovým prostředím. V předmětu Počítačové sítě jsme se učili pracovat se síťovými prvky značky Cisco, což je i vybavení aplikované napříč infrastrukturou sítí Tieto. Takto pokud jsem potřeboval kontaktovat síťový tým a spolupracovat, věděl jsem jak tyto mechanismy fungují.
Další užitečný předmět byl Operační systémy. V tomto předmětu jsme se učili, jak programovat v jazyce C přímo pro operační systém Linux. To mi též umožnilo více proniknout do chodu systému, který jsem v rámci své praxe spravoval.

7.2 Scházející znalosti

Většinu znalostí operačního systému Linux jsem získal samostudiem v průběhu dlouholetého používání tohoto systému. Za dobu používání Linuxu jsem vystřídal několik různých distribucí, což mi též přineslo větší rozhled. Stále mi ale chybí hlubší znalosti vnitřních systémových pochodů, které by mi pomohly plně porozumět tomuto minoritnímu operačnmu systému.

8 Závěr

Jak jsem se v úvodu zmínil, jedná se o mezinárodní společnost, která má spoustu zaměstnanců a také spoustu zákazníků a naprostý individualismus v ní nemá místo, protože skupina je silnější, než jednotlivec. Proto také různé problémy řeší různé týmy a spoustu problémů Unix team ani vyřešit nemůže, ať už z důvodu, že nemáme dostatečná přístupová práva, nebo protože zjistíme, že problém není systémového charakteru, ale např. aplikačního. Na tyto problémy se zase specializuje jiný tým. Řešení spousty problémů navíc často bývá banální a nejtěžší částí tak bývá zdroj a z něj plynoucí řešení problému odhalit.

Během své praxe ve společnosti Tieto jsem se naučil hodně věcí z reálného firemního prostředí. Jako obrovský přínos považuji skutečnost, že jsem se mohl setkat se systémy a zařízeními, na kterých beží. S většinou z těchto systémů, ať už jde o operační systémy HP-UX, Solaris, AIX nebo HP Blade servery, bych se totiž nejspíše jinde nesetkal. Taky jsem si mohl vyzkoušet, jak probíhá týmová práce ve velké společnosti, kterou Tieto bezpochyby je. Kolegové mi byli po celou dobu mé praxe nápomocni, stejně jako jsem já nejednou znamenal přínos v podobě originálních nápadů a svěžího pohledu na věc. Pokud mi kolegové nebyli schopni poradit, nalezl jsem spoustu informací např. na českých portálech root.cz[6] nebo abclinuxu.cz[4]. Celou dobu jsem měl taky k dispozici několik knih, ze kterých jsem využil především Kompletní příručku administrátora[1].

Reference

[1] Evi Nemeth, Garth Snyder, Trent R. Hein; Překlad: David Vozák, Miloš Průdek, Lubomír Ptáček, Jiří Huf, Jiří Berka; *Kompletní příručka administrátora*. Computer Press, a.s., Vydání první, 2008.

[2] http://www.tieto.cz/o-nas/historie/ [online], *Historie společnosti Tieto*.

[3] http://cs.wikipedia.org/wiki/RAID [online], *RAID - Wikipedie*

[4] http://www.abclinuxu.cz/ [online], *AbcLinuxu.cz*

[5] http://www.root.cz/clanky/lvm-prakticke-ukazky/ [online] *LVM: Praktické ukázky*

[6] http://www.root.cz/ [online], *Root.cz*